SQL for Data Analytics Interviews

User Guide: Setting Up the Retail Sales Dataset

This guide provides step-by-step instructions for creating a database table and importing the Retail Sales Dataset using SQL.

1. Setting Up the Database

1.1 Install PostgreSQL and pgAdmin

- Download and install PostgreSQL from postgresql.org.
- Install pgAdmin for managing the database via a graphical interface.

1.2 Connect to the Database

1. Open pgAdmin and log in with your PostgreSQL credentials.
2. Create a new database:
    - Right-click on "Databases" > "Create" > "Database."
    - Enter a name (e.g., `RetailSalesDB`) and save.

---

2. Creating the Table

2.1 Define the Table Schema

Run the following SQL query in pgAdmin to create the table:

```
CREATE TABLE RetailSales (
    TransactionID INT PRIMARY KEY,
    Date DATE NOT NULL,
    CustomerID VARCHAR(50) NOT NULL,
    Gender VARCHAR(10) NOT NULL,
    Age INT NOT NULL,
    ProductCategory VARCHAR(100) NOT NULL,
```

```
    Quantity INT NOT NULL,
    PricePerUnit INT NOT NULL,
    TotalAmount INT NOT NULL
);
```

- **TransactionID**: Unique identifier for each transaction.
- **Date**: The date of the transaction in `YYYY-MM-DD` format.
- **CustomerID**: Unique identifier for each customer.
- **Gender**: Gender of the customer (e.g., Male, Female).
- **Age**: Age of the customer in years.
- **ProductCategory**: Category of the product purchased.
- **Quantity**: Number of units purchased.
- **PricePerUnit**: Price of one unit of the product.
- **TotalAmount**: Total cost of the transaction.

---

3. Importing the Dataset

3.1 Prepare the Dataset

1. Download the dataset from this link - https://drive.google.com/file/d/1OyUpe_EPnNvg9RCX7gHuHci2w7dS5chb/view?usp=sharing and place it in an accessible location (e.g., `/path/to/retail_sales_dataset.csv`).

3.2 Import the Data

Using pgAdmin

1. Open the Query Tool in pgAdmin.
2. Run the following SQL command to import the dataset:

```
COPY RetailSales FROM '/path/to/retail_sales_dataset.csv' DELIMITER ','
CSV HEADER;
```

- Replace `/path/to/retail_sales_dataset.csv` with the actual path to the file.
- `DELIMITER ','`: Specifies that the file is comma-separated.
- `CSV HEADER`: Skips the first row (header).

---

4. Verifying the Data

Run the following query in the Query Tool to verify the number of rows imported:

```
SELECT COUNT(*) FROM RetailSales;
```

Ensure the result matches the expected number of rows in the dataset which is 1000 records.

5. Explore the dataset

After you are finished importing the dataset, don't stop there. Try to just explore the dataset and try to understand the business value, the impact it can create and what kind of questions can be answered using the dataset?

6. Creating a new table to demonstrate joins

```
-- 1. Create a secondary table for demonstrating joins
CREATE TABLE customer_details (
    customer_id VARCHAR(10) PRIMARY KEY,
    customer_name VARCHAR(50),
    city VARCHAR(50),
    loyalty_points INT
);

-- 2. Insert sample data into the secondary table
INSERT INTO customer_details (customer_id, customer_name, city,
loyalty_points) VALUES
('CUST001', 'John Doe', 'Toronto', 500),
('CUST002', 'Jane Smith', 'Montreal', 1200),
('CUST003', 'Robert Brown', 'Vancouver', 300),
('CUST004', 'Michael Davis', 'Calgary', 800),
('CUST005', 'William Wilson', 'Ottawa', 150),
('CUST006', 'Emily Johnson', 'Edmonton', 950),
('CUST007', 'Sophia Martinez', 'Winnipeg', 1100),
('CUST008', 'Olivia Anderson', 'Quebec City', 600),
('CUST009', 'James Taylor', 'Halifax', 700),
('CUST010', 'Emma Moore', 'Victoria', 400);
```

7. Solve these questions

1. Find the total sales amount for each product category.

2. Which city has the highest total loyalty points?

3. Find the top 3 customers who spent the most.

4. Calculate the average age of customers by gender.

5. Retrieve all transactions where customers spent more than 1000.

6. List customers and their loyalty points for all transactions.

7. What is the average quantity sold per product category?

8. Find the total sales amount for each city.

9. Which product category has the highest average unit price?

10. Show all customers who have more than 500 loyalty points.

11. Rank customers by total spending.

12. Identify the most popular product category based on quantity sold.

13. Find the cumulative sales for each customer.

14. Calculate the difference in loyalty points between the highest and lowest in each city.

15. Retrieve customers who made purchases in more than 1 product category.

8. Solutions

```sql
-- Question 1: Find the total sales amount for each product category.
SELECT product_category, SUM(total_amount) AS total_sales
FROM retail_sales
GROUP BY product_category;

-- Question 2: Which city has the highest total loyalty points?
SELECT city, SUM(loyalty_points) AS total_loyalty_points
FROM customer_details
GROUP BY city
ORDER BY total_loyalty_points DESC
LIMIT 1;

-- Question 3: Find the top 3 customers who spent the most.
SELECT customer_id, SUM(total_amount) AS total_spent
FROM retail_sales
GROUP BY customer_id
ORDER BY total_spent DESC
LIMIT 3;

-- Question 4: Calculate the average age of customers by gender.
SELECT gender, AVG(age) AS avg_age
FROM retail_sales
GROUP BY gender;

-- Question 5: Retrieve all transactions where customers spent more than
1000.
SELECT *
FROM retail_sales
WHERE total_amount > 1000;

-- Question 6: List customers and their loyalty points for all
transactions.
SELECT rs.customer_id, rs.total_amount, cd.loyalty_points
FROM retail_sales rs
LEFT JOIN customer_details cd
ON rs.customer_id = cd.customer_id;

-- Question 7: What is the average quantity sold per product category?
SELECT product_category, AVG(quantity) AS avg_quantity
FROM retail_sales
GROUP BY product_category;

-- Question 8: Find the total sales amount for each city.
SELECT cd.city, SUM(rs.total_amount) AS city_sales
```

```sql
FROM retail_sales rs
JOIN customer_details cd
ON rs.customer_id = cd.customer_id
GROUP BY cd.city;

-- Question 9: Which product category has the highest average unit
price?
SELECT product_category, AVG(price_per_unit) AS avg_unit_price
FROM retail_sales
GROUP BY product_category
ORDER BY avg_unit_price DESC
LIMIT 1;

-- Question 10: Show all customers who have more than 500 loyalty
points.
SELECT *
FROM customer_details
WHERE loyalty_points > 500;

-- Question 11: Rank customers by total spending.
SELECT customer_id, SUM(total_amount) AS total_spent,
       RANK() OVER (ORDER BY SUM(total_amount) DESC) AS rank
FROM retail_sales
GROUP BY customer_id;

-- Question 12: Identify the most popular product category based on
quantity sold.
SELECT product_category, SUM(quantity) AS total_quantity
FROM retail_sales
GROUP BY product_category
ORDER BY total_quantity DESC
LIMIT 1;

-- Question 13: Find the cumulative sales for each customer.
SELECT customer_id, total_amount,
       SUM(total_amount) OVER (PARTITION BY customer_id ORDER BY
transaction_id) AS cumulative_sales
FROM retail_sales;

-- Question 14: Calculate the difference in loyalty points between the
highest and lowest in each city.
SELECT city, MAX(loyalty_points) - MIN(loyalty_points) AS points_diff
FROM customer_details
GROUP BY city;

-- Question 15: Retrieve customers who made purchases in more than 1
```

```
product category.
SELECT customer_id
FROM retail_sales
GROUP BY customer_id
HAVING COUNT(DISTINCT product_category) > 1;
```